Trading efficiency for effectiveness in similarity-based indexing for image databases *

Julio Barros

University of Virginia
Los Alamos National Laboratory

James French, Worthy Martin

Computer Science Dept.
University of Virginia
Charlottesville, Va 22903

Patrick Kelly

Los Alamos National Laboratory
Los Alamos, NM 87545

ABSTRACT

Image databases typically manage feature data that can be viewed as points in a feature space. Some features, however, can be better expressed as a collection of points or described by a probability distribution function (PDF) rather than as a single point. In earlier work we introduced a similarity measure and a method for indexing and searching the PDF descriptions of these items that guarantees an answer equivalent to sequential search. Unfortunately, certain properties of the data can restrict the efficiency of that method. In this paper we extend that work and examine trade-offs between efficiency and answer quality or effectiveness. These trade-offs reduce the amount of work required during a search by reducing the number of undesired items fetched without excluding an excessive number of the desired ones.

Keywords: image databases, similarity-based retrieval, efficiency and effectiveness trade-offs, similarity measures

1 INTRODUCTION

Many image databases[7,8,3] manage feature data that can be viewed as points in a feature space. In these systems similarity is typically measured by the (weighted) Euclidean or city-block distance. Queries are efficiently processed by using access methods, such as B-trees[4] or R-trees,[6] to find the closest feature points to a query point.

*Appeared in IS&T/SPIE East - Digital Image Storage and Archiving Systems, Oct 26-27, Philadelphia, PA.

Some features, however, can be better expressed as a collection of points rather than as a single point. In these cases probability density functions (PDFs) can often be used to summarize or approximate the underlying data. PDFs are also used to describe both collections of measurements and measurements with known errors. For example, once objects have been segmented from an image, the color or texture values of the constituent pixels can be summarized by their means and standard deviations.

Composite descriptions, such as PDFs, are typically searched sequentially or are simplified considerably. Sequential search quickly becomes infeasible as the database size increases and the feature effectiveness diminishes as descriptions are over simplified. In earlier work[1,2] we described a similarity measure and a method of indexing and searching the PDF descriptions that guarantees an answer equivalent to one arrived at by sequential search. Unfortunately, certain properties of the data can severely restrict the efficiency of that method. In this paper we extend previous work and examine trade-offs between efficiency and answer quality or effectiveness. These trade-offs reduce the number of undesired items fetched without excluding an inordinate number of the desired ones.

Section 2 discusses the background of the problem. Section 3 explores issues in increasing efficiency without incurring unacceptable sacrifices in effectiveness. It also discusses the effects on performance using actual data sets. Section 4 discusses the conclusions of this work.

2 BACKGROUND

In our system a collection of points or items is described by a probability density function (PDF) represented by a pair of real numbers, $(\bar{X}, s)$, corresponding to the mean and standard deviation of the distribution. Given a large collection of items or item records we seek a way to organize them that allows efficient searches. In a typical search the user specifies a query point and wishes to find collections of points whose distributions indicate a high similarity with the query point. PDFs are inherently different from simple point data and range/interval data due to their individual standard deviation. Consequently, the similarity measures and access methods used need to be tailored to the specific characteristics of the PDF description.

Although Euclidean distance works well in many applications, we would like to use a similarity measure that takes into account the individual item variance. For example, in Figure 1, the query point is approximately the same Euclidean distance away from the mean of item A and item B. However, item A is more likely to contain the query point and we would like it to be judged more similar to the query point than item B.

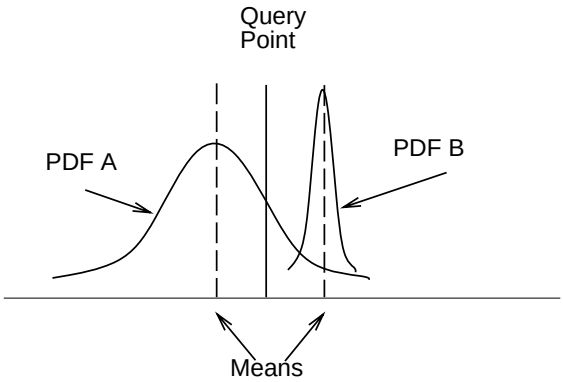

Figure 1: Variance plays an important role in similarity judgments.

The importance of variance is captured with a normalized distance measure. In Equation 1 the distance

between a query point and an item mean is normalized by the item's standard deviation. With this distance or *dissimilarity* measure the greater the score the less similar the item and query point. The dissimilarity measure we use is:

$$\frac{\left|\bar{X}_i - q\right|}{s_i} = M_i \tag{1}$$

Where q is the query point, $\bar{X}_i$ is the mean of the feature of the ith item, s_i is the standard deviation of the feature of the ith item, and M_i is the number of standard deviations between the query point and the mean of the item. We will abbreviate the Euclidean distance between the query point and item mean, $\left|\bar{X}_i - q\right|$, as d_i.

This measure may at first seem similar to a weighted Euclidean measure. In a weighted Euclidean measure the importance of individual feature components is emphasized by scaling the Euclidean distances among those components. The same scale factor is used when comparing the query point to every record in the database. In our method the Euclidean distances are also scaled during the comparison. However, each item standard deviation specifies the possibly unique scale factor to use when comparing that item to a query point.

To process a query a user specifies a query point, q, and a threshold value, M. Each item, i, in the database is examined to determine if $M_i \leq M$. The *complete answer set* contains all items that meet this *similarity criterion.* After *all* the items are examined the complete answer set is returned to the user. The similarity criterion is:

$$\left\{ i \;\middle|\; \frac{\left|\bar{X}_i - q\right|}{s_i} \leq M \right\} \tag{2}$$

As presented so far, we must apply the similarity criterion sequentially to each item in the database. However, this test can be modified to be more efficient in many situations. The efficiency can be realized through an indexing structure that yields the complete answer set without directly calculating M_i for each item. This is possible when we rewrite the the similarity criterion as:

$$q - Ms_i \leq \bar{X}_i \leq q + Ms_i \tag{3}$$

In this form we notice that the item mean $\bar{X}_i$ must fall in the range between $q - Ms_i$ and $q + Ms_i$. The query point q specifies the center of the range and M and s_i specify the size of the range.

This process may seem similar to traditional range or interval searches. However, the key difference is that the extent of the ranges depends on the user supplied threshold value M. Consequently the range extents are not fixed or known before query time. If a fixed value for M is to be used for all queries then other methods such as interval trees[5] can be used.

The values for q and M are different for each query but are fixed at the beginning of the each query. Unfortunately, both $\bar{X}_i$ and s_i depend on the particular item being examined. A crude filter, Equation 4, is created by using a constant, s', in place of the individual s_i.

$$\left\{ i \,\middle|\, q - Ms' \leq \bar{X}_i \leq q + Ms' \right\} \tag{4}$$

This bounds the ends of the *query range* at the beginning of the query. If $s' = s_{max}$, the largest s_i in the database, the query range is guaranteed to contain all items that could possibly pass the similarity criterion. It will, most likely, also contain other items.

To take advantage of this filter some prequery processing is required. We make note of s_{max}, the largest item standard deviation, and store the item means in a data structure that allows efficient range searches. Queries are then processed in two steps as depicted in Figure 2. The first step calculates the query range and retrieves the *candidate set*, all the items with means in that range. The second step applies the similarity criterion to each of the items in the candidate set to extract the answer set. It is important to observe that since $s' = s_{max}$ is used for Equation 4, the answer set returned by this two stage process is guaranteed to be the same as the complete answer set that is returned by the sequential method.

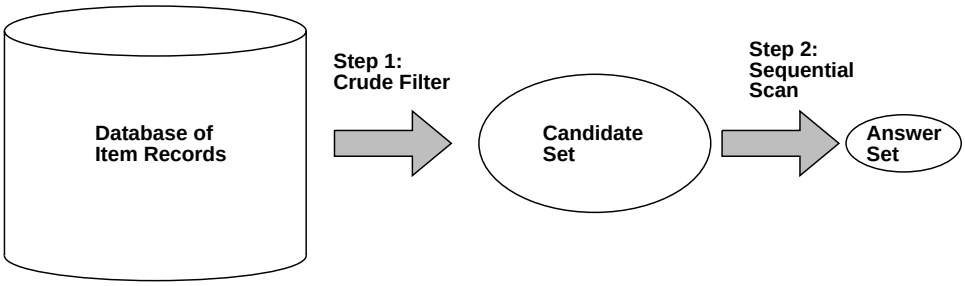

Figure 2: Query processing in two steps.

3 EFFECTIVENESS AND EFFICIENCY

The efficiency realized by the process described in Section 2 depends on the size of the candidate set returned by the calculated range search and the degree to which the cost of creating and maintaining the indexing structure can be amortized across multiple queries. If the candidate set returned includes all items in the database, then we have only incurred additional overhead. However, considerable savings may be realized if this set is significantly smaller than the database. The size of the candidate set, and thus the efficiency of the procedure, is affected by two factors.

1. The first factor is the relative positions of the means and queries in the feature space. If the means are concentrated in a small area, then all queries in this area will retrieve a large percentage of the database.
2. The second factor is the magnitude of the largest standard deviation. If there is even one PDF description with a large standard deviation the query range will be wide and will include an excessive portion of the database.

The best performance is realized by a database with a relatively small s_{max}, with means and queries that are uniformly distributed throughout the space, and with a high number of queries that can take advantage of the indexing mechanism.

Since little can be done to control the positions of the data or queries, little can be done to counter the effects of the first factor. However, to reduce the effects of the second factor it may be beneficial to use a smaller value for s' than s_{max} in Equation 4. The smaller value relaxes the guarantee that the candidate range will contain the complete answer set. However, if the loss of a few answer items is acceptable, significant increases in efficiency are possible. In this case the answer set will be a subset of the complete answer set. The remainder of this paper focuses on the effects on effectiveness and efficiency of reducing the value of s'.

3.1 MEASURING AND ESTIMATING PERFORMANCE

We evaluate the performance of an index along two dimension: *efficiency* and *effectiveness*. Efficiency is concerned with how much work is performed to answer a query. Effectiveness is concerned with how well a query is answered.

To measure efficiency we use the *comparison percentage*. The comparison percentage is the percentage of the database that is individually compared to the query point using the similarity criterion. Only the items returned by the filter are compared, so the comparison percentage is the size of the candidate set divided by N, the size of

the database.

$$\text{comparison percentage} = \frac{|candidate\_set|}{N} \tag{5}$$

The expected comparison percentage depends on the location and size of the query range and the density of item means in that region. Two different query points with the same size query range will result in candidate sets of different size depending on the density of items at the different locations.

For example, if the queries and item means are distributed uniformly throughout the feature space the expected comparison percentage is directly related to the size of the query range. In this case, the percentage of means retrieved is equivalent to the percentage of the space covered by the query range. If the query range is 32 units wide and the feature space 256 units wide, approximately one eighth of the items in the database will be retrieved. The actual percentage will be less for queries near the boundaries of the feature space since the effective width of the query range will be smaller.

The effectiveness of the answer is measured by *recall* and *precision*. Recall is a measure of the completeness of the answer given to the user. Recall is the percentage of desired items returned or the the size of the answer set divided by the size of the complete answer set.

$$\text{Recall} = \frac{|answer\_set|}{|complete\_answer\_set|} \tag{6}$$

As the value of s' is reduced and more items are excluded from the candidate set the chance of an item being excluded from the answer set increases. Consequently, recall is expected to drop as s' is reduced.

Items are excluded based on their Euclidean distance, d_i to the query point. Items that are further away will be excluded first. If one of these excluded items is an answer item we know it must have had a large standard deviation to meet the similarity criterion ($d_i/s_i \leq M$). Reducing s' will exclude answer items only if the standard deviation of the answer item is less than s'. Consequently, the only answer items at risk of being excluded are those with standard deviations larger than s'. Therefore, the rate at which recall decreases depends on the number of standard deviations that are less than s'. For example, if s_{max} were considerably larger than most of the standard deviations in the database, s' could be reduced considerably and still be larger than most standard deviations. This would result in a lower chance of excluding an answer item from the candidate set and result in a higher recall percentage.

However, if all standard deviations were (roughly) equal, an s' less than s_{max} would result in many items with standard deviations greater than s'. This would increase the chance that an answer item would be excluded from the candidate set. This type of standard deviation distribution is not common in the data we have been using. Typically, most standard deviations will be much smaller than the maximum standard deviation.

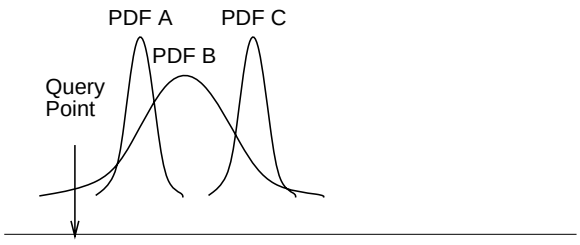

Figure 3: Items are excluded based on Euclidean distance not the similarity score.

It is interesting to note that the strategy of using values for s' less than s_{max} tends to exclude answer items with large standard deviations. The answer items with smaller standard deviations, which are often thought of as better defined and more desirable, are included in the candidate set. Note that items are excluded based on their Euclidean distance to the query point and that this does not imply that items are excluded according to

their dissimilarity score (Equation 1). For example, Figure 3 shows items A, B, and C with standard deviations $s_B = 3s_A$ and $s_C = s_A$, and with Euclidean distance between the mean and a particular query point $d_B = 2d_A$ and $d_C = 3d_A$. This implies that the similarity scores ($d/s = M$), $M_B = 2/3M_A$ and $M_C = 3M_A$. Since a lower score signifies a better match, the query point is more similar to item B than A and more similar to A than C. As the magnitude of s' is reduced, item C will be excluded before item A or B. However, item B will be excluded from the candidate set before item A even though M_B is a lower score than M_A.

Precision is a measure of the quality of the intermediate or candidate set. In this technique precision gives us an idea of how well our crude filter, Equation 4, is performing. Precision is the percentage of the candidate set that belongs in the answer set.

$$\text{Precision} = \frac{|answer\_set|}{|candidate\_set|} \tag{7}$$

As the value of s' is decreased, items at a greater Euclidean distance from the query point will be excluded. The items remaining in the candidate set are at a smaller Euclidean distance from the query point and will meet the similarity criterion with smaller values of s_i. Since the acceptable range of values for s_i increases, the candidate set items are in general more likely to pass the similarity criterion. Therefore, as s' is reduced it becomes increasingly likely that the items remaining in the candidate set will meet the similarity criterion and the precision of the candidate set will tend to increase. For example, if the Euclidean distances under consideration are very large, few standard deviations may be large enough to meet the similarity criterion ($d_i/s_i \leq M$). However, if the distances are very small, a large percentage of the individual s_i's are likely to be large enough to meet the similarity criterion. The rate at which precision increases depends on the distribution of the standard deviations and is not likely to increase monotonically, but rather will tend to increase probabilistically.

To gauge the performance of our indexing and query processing methods we must jointly consider the three measures: recall, precision, and comparison percentage. For example, suppose we have a database of 500 records and that 100 of these items pass the similarity criterion for a particular query. If during the query processing the filter extracts 250 records, 75 of which are returned in the answer set, we have a recall of 75%, precision of 30%, and a comparison percentage of 50%,

In practice it may be difficult or impossible to predict the number or distribution of query points. It may also be difficult to describe or approximate the distribution of means and standard deviations. In this case it is necessary to make empirical performance measurements. These measurements can be studied to help reveal trade-offs affecting performance. An analysis of performance on both synthetic and real data is given in the following section.

3.2 EXPERIMENTAL RESULTS

To test the practical effects of reducing s' we conducted several retrieval experiments. In each experiment 200 different values of s' were used. The values of s' were calculated in regular intervals between s_{max} and 0. An s' value of 0 implies that a query point and a mean were similar only if they were equal. For each value of s', 100 different queries were run using a random query point chosen from the range of the means of the particular database. The numbers reported below are the averages for the 100 queries. The threshold value, M, was 2.

For the first experiment, Figure 4 and Table 1, 100,000 items were systematically generated. The means of the items were real numbers uniformly distributed in the range 0 and 256. The standard deviations were real numbers uniformly distributed in the range 0 to 50.

If a value of s_{max} is used a typical query was expected to have 100% recall. The size of the query ranges varied between 200 ($q \pm M * s_{max} = q \pm 50 * 2$) and 100 at the edges of the feature space and covered between 39% and 78% of the database. On average the query ranges covered approximately 63% of the feature space. Consequently, the comparison rate is expected to be approximately 63%. Since the queries and means are uniformly distributed

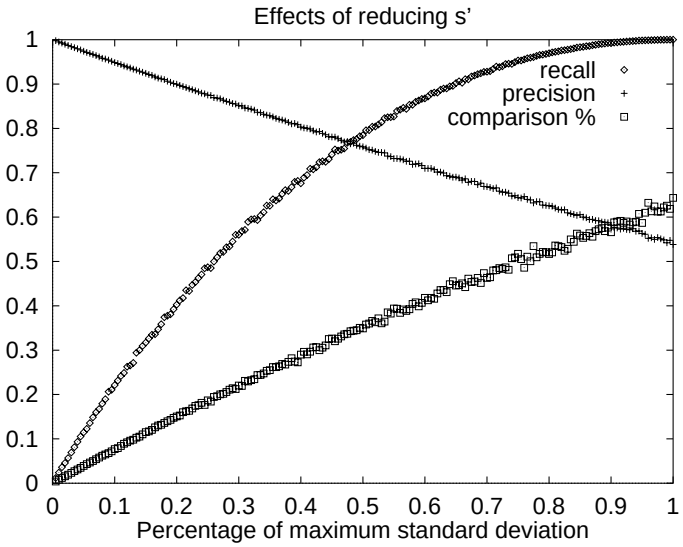

Figure 4: Performance of synthetic uniform data

the comparison percentage is expected to drop off linearly and the precision is expected to increase linearly as s' is decreased.

For the synthetic data an s' value of s_{max} results in 100% recall, 54% precision, and a 64% comparison percentage. An s' value of $1/2s_{max}$ results in 78% recall, 76% precision, and a 35% comparison percentage. The performance results of this experiment are shown in Figure 4 and Table 1. A linear decrease in the value of s' resulted in a linear decrease in comparison percentage and a linear increase in candidate set precision. Recall is not linear but decreases noticeably early in the experiment.

	Synthetic Data			Color Data		
% of s_{max}	Recall %	Precision %	Comparison %	Recall %	Precision %	Comparison %
100	100	54	64	100	31	57
90	99	58	57	99	34	51
80	97	63	52	99	37	45
70	93	67	46	99	41	41
60	87	71	42	99	46	37
50	78	76	35	97	55	30
40	68	80	29	93	65	24
30	56	85	22	84	75	20
20	40	90	15	65	89	13
10	22	95	08	38	97	07

Table 1: Performance results for synthetic and color data

The second experiment, Figures 5, 6, and Table 1, uses the color information from 310 images of various types. The color information in each image was clustered into 10 clusters using a k-means algorithm. The clustering step is similar to color quantization. Although the clustering is done on the red, green, and blue bands simultaneously, queries are currently processed on a single band at a time. The data for the red band was used in this experiment. The means of the clusters were in the range 0-255 and standard deviations in the range 0-44. Figure 5 shows the histogram of the positions of the means and of the standard deviations. Note that the means are neither overly concentrated in one area or uniformly distributed across the space. The standard deviations are also not

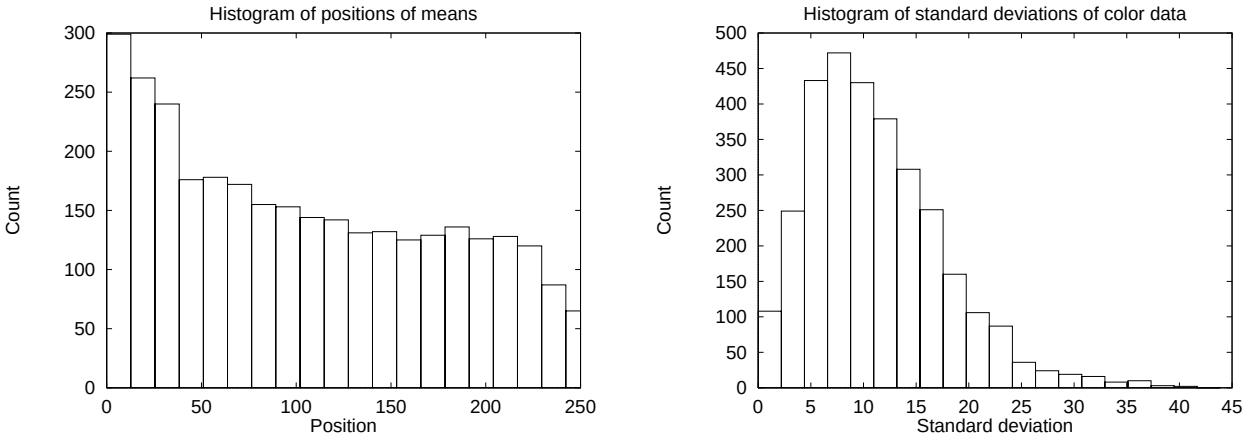

Figure 5: Histograms of color data

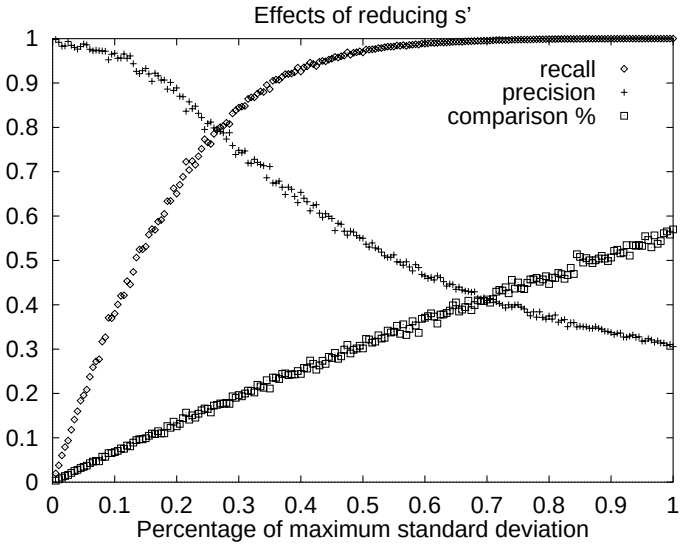

Figure 6: Performance of color data

uniformly distributed between 0 and the maximum.

Since the distributions of the means and standard deviations are not linear and are difficult to estimate it is difficult to estimate the expected performance levels. However, since most of the s_i are much smaller than s_{max}, recall percentage is expected to stay high even when s' is substantially reduced. Comparison percentage is expected to decrease linearly with a linear reduction in s'.

For the color data (see Table 1) experiment an s' value of s_{max} results in 100% recall, 31% precision and a 57% comparison percentage. As s' is reduced the comparison percentage decreases linearly even though recall is initially not severely affected. For example, an s' value of $1/2 s_{max}$ results in 97% recall, 55% precision, and a 30% comparison percentage. Note that by reducing s' to 30% of s_{max}, 84% of the complete answer set can be retrieved by examining only 20% of the entire database.

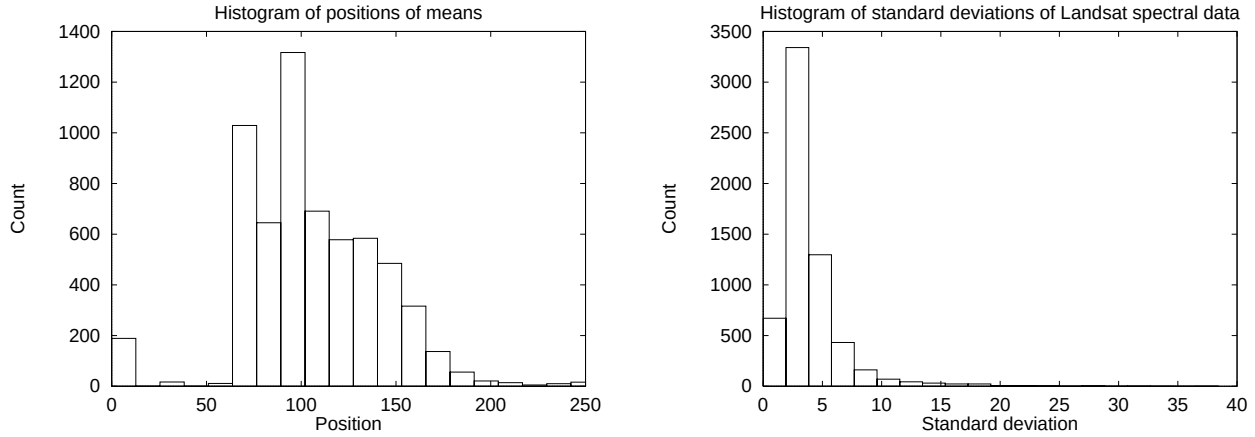

Figure 7: Histograms of Landsat spectral data

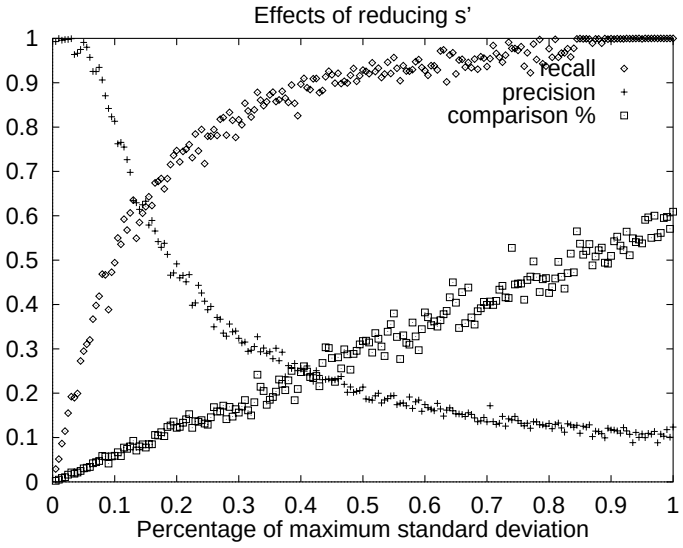

Figure 8: Performance of Landsat spectral data

The final two experiments, Figures 7, 8, 9 and 10 and Table 2, were performed on one band of multi-band Landsat data and on one band of multi-band texture data. The Landsat data was extracted from images of Moscow, Cairo, Los Alamos, and Albuquerque. Six bands were used in the clustering step. Only the first band was indexed and used in this experiment. The texture data was generated by using the Laws Texture measures

	Landsat Data			Texture Data		
% of s_{max}	Recall %	Precision %	Comparison %	Recall %	Precision %	Comparison %
100	100	12	61	100	04	65
90	99	12	51	99	05	60
80	98	13	43	93	04	45
70	94	14	40	86	06	47
60	93	17	38	81	06	41
50	93	21	32	68	06	28
40	89	26	25	55	09	20
30	82	32	16	48	09	24
20	72	46	12	33	08	09
10	49	82	06	23	17	06

Table 2: Performance results for Landsat and texture data

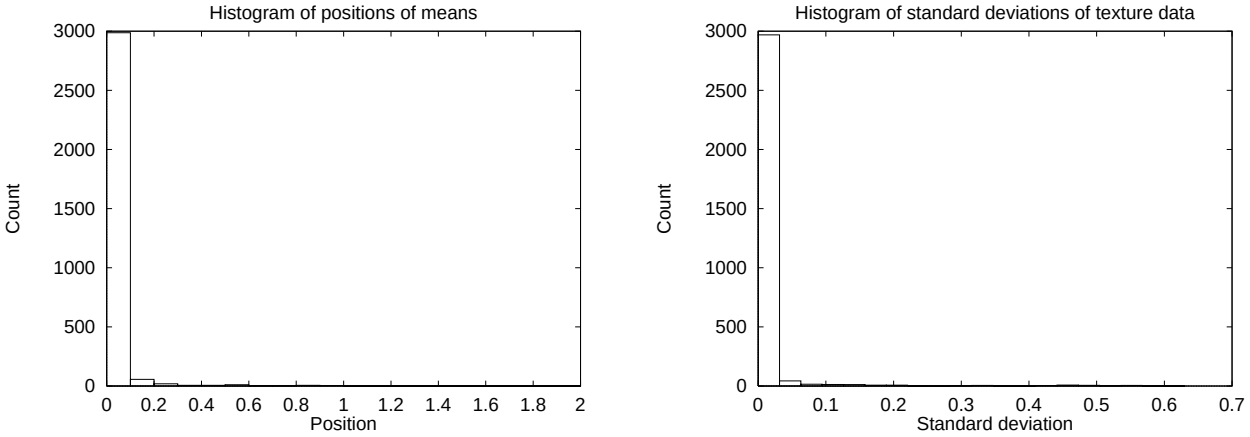

Figure 9: Histograms of texture data

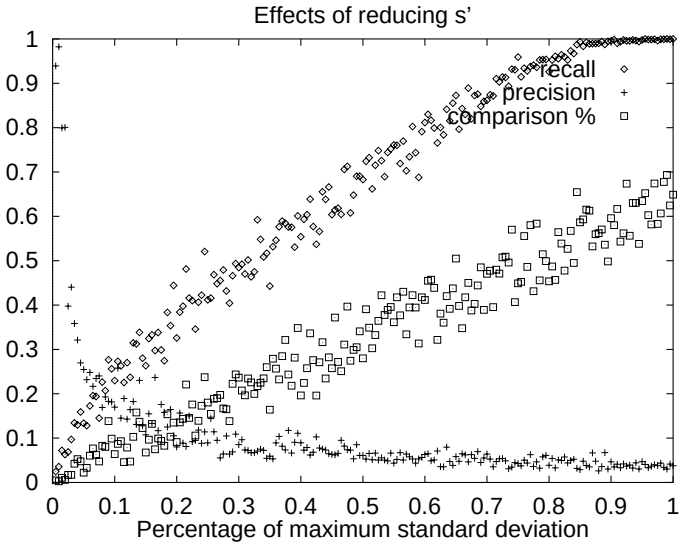

Figure 10: Performance of texture data

on the images of second experiment. The 4-dimensional texture coefficients were then clustered into 10 clusters. Only the first dimension is used in the queries.

In Figures 7 and 9 we see that the locations of the means and values of the standard deviations are somewhat concentrated for the landsat data and very concentrated for the texture data. This concentration leads to more variation in the performance results. Although the average comparison percentage remains comparable to the previous experiments the precision is considerably lower. Also, in the landsat data recall has the same characteristic curve as the previous experiments though in the texture data recall performance drops nearly linearly.

4 CONCLUSION

In earlier work[1,2] we stated that a probability density function (PDF) can be used to describe a collection of points or measurements with known errors. We introduced a similarity measure that can be used in similarity based retrieval of these descriptions. Additionally, we described a method of indexing and searching collections of PDF descriptions that guarantees an answer equivalent to sequential search. Unfortunately, certain properties of the data can severely restrict the efficiency of that method. In this paper we have extended previous work and examined trade-offs between efficiency and answer quality, effectiveness.

In this paper we showed that these trade-offs reduce the amount of work required to process a query by reducing the number of undesired items without excluding an excessive number of the desired items. We give four examples where 93% of the complete answer set is retrieved by examining only 46%, 24%, 38%, and 45% of the database. Additionally, examining only 6-8% of the database can retrieve 22%, 38%, 49%, and 23% of the complete answer set. Efficiency gains in large databases can be obtained with out excessively degrading retrieval effectiveness.

5 ACKNOWLEDGMENTS

Portions of this work were were performed under a U.S. Government contract (W-7405-ENG-36) by Los Alamos National Laboratory, which is operated by the University of California for the U.S. Department of Energy. This work was also supported by an NSF Graduate Engineering Education fellowship.

6 REFERENCES

[1] Julio Barros, James French, Worthy Martin, and Patrick Kelly. System for indexing multi-spectral satellite images for efficient content-based retrieval. In *Proceedings of IS&T/SPIE: Storage and Retrieval for image and Video Databases III*, San Jose, California, February 1995.

[2] Julio Barros, James French, Worthy Martin, Patrick Kelly, and James M. White. Indexing multispectral images for content-based retrieval. In *Proceedings of the 23rd AIPR Workshop on Image and Information Systems: Applications and Opportunities*, Washington, D.C., October 1994.

[3] Tzi-cker Chiueh. Content-based image indexing. In *Proceedings of the 20th VLDB Conference*, Santiago, Chile, 1994.

[4] Douglas Comer. The ubiquitous B-tree. *Computing Surveys*, 11(2), 1979.

[5] Thomas H. Cormen, Charles E. Leirserson, and Ronald L. Rivest. *Introduction to Algorithms*. The MIT Press, 1990.

[6] Antonin Guttman. R-tree: A dynamic index structure for spatial searching. *ACM SIG-MOD Proc.*, 1984.

[7] W. Niblack, R. Barber, W. Equitz, M. Flickner, E. Glasman, D. Petkovic, P. Yanker, C. Faloutsos, and G. Taubin. The QBIC project: Querying images by content using color, texture and shape. In Wayne Niblack, editor, *Storage and Retrieval for Image and Video Databases*, pages 173–187, Bellingham, Washington, February 1993. SPIE, SPIE.

[8] N. Yazdani, M. Ozsoyoglu, and G. Ozsoyoglu. A framework for feature-based indexing for spatial databases. In James C. French and Hans Hinterberger, editors, *7th Int. Working Conference on Scientific and Statistical Database Management*. IEEE Computer Society Press, September 1994.